Overview

Introduction
Modelling parallel systems
Linear Time Properties
Regular Properties
Linear Temporal Logic (LTL)
Computation-Tree Logic

Equivalences and Abstraction
  bisimulation
  CTL, CTL*-equivalence
  computing the bisimulation quotient
  abstraction stutter steps
  simulation relations
Classification of implementation relations
Classification of implementation relations

- **linear vs. branching time**
  - linear time: trace relations
  - branching time: (bi)simulation relations

- **(nonsymmetric) preorders vs. equivalences**:
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation

- **strong vs. weak relations**
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps
Classification of implementation relations

- **linear vs. branching time**
  - linear time: trace relations
  - branching time: (bi)simulation relations

- **(nonsymmetric) preorders vs. equivalences:**
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation

- **strong vs. weak relations**
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps
Classification of implementation relations

- **linear vs. branching time**
  - linear time: trace relations
  - branching time: (bi)simulation relations

- **(nonsymmetric) preorders vs. equivalences:**
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation

- **strong vs. weak relations**
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps
The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion
- “unidirected” version of bisimulation:
The simulation preorder is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion
- "unidirected" version of bisimulation:

if $\mathcal{T}_1$ is simulated by $\mathcal{T}_2$ then $\mathcal{T}_2$ can mimick all steps of $\mathcal{T}_1$, but possibly has more behaviors
The simulation preorder

is a nonsymmetric branching time relation
- plays of central role for abstraction
- the BT-analogue to trace inclusion
- “unidirected” version of bisimulation:
  
  if $T_1$ is simulated by $T_2$ then $T_2$ can mimick all steps of $T_1$, but possibly has more behaviors

- relies on a coinductive definition
  (as bisimulation equivalence)

*here:* just strong simulation, i.e., no abstraction from stutter steps
Simulation for two TS

let $\mathcal{T}_1 = (S_1, Act_1, \rightarrow_1, S_{0,1}, AP, L_1)$

$\mathcal{T}_2 = (S_2, Act_2, \rightarrow_2, S_{0,2}, AP, L_2)$

be two transition systems

- over the same set $AP$ of atomic propositions
- possibly with terminal states
Simulation for a pair of TS

Simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

1. If $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$

2. For all $(s_1, s_2) \in \mathcal{R}$:
   \[\forall s'_1 \in \text{Post}(s_1) \ \exists s'_2 \in \text{Post}(s_2) \text{ s.t. } (s'_1, s'_2) \in \mathcal{R}\]

(I) For all initial states $s_1$ of $\mathcal{T}_1$ there is an initial state $s_2$ of $\mathcal{T}_2$ with $(s_1, s_2) \in \mathcal{R}$
Simulation preorder ≤

simulation for $(T_1, T_2)$: relation $R \subseteq S_1 \times S_2$ s.t.

(1) labeling condition
(2) stepwise simulation condition
(I) initial condition

simulation preorder ≤ for TS:

$T_1 \preceq T_2$ iff there exists a simulation $R$
for $(T_1, T_2)$
Simulation preorder $\preceq$

Simulation for $(T_1, T_2)$: relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

1. Labeling condition
2. Stepwise simulation condition
3. Initial condition

<table>
<thead>
<tr>
<th>simulation preorder $\preceq$ for TS:</th>
</tr>
</thead>
</table>
|$T_1 \preceq T_2$ iff \{
\begin{align*}
\text{there exists a simulation } \mathcal{R} \\
\text{for } (T_1, T_2)
\end{align*}
|}

If $s_1$ is a state of $T_1$ and $s_2$ a state of $T_2$ then

$s_1 \preceq s_2$ iff there exists a simulation $\mathcal{R}$ for $(T_1, T_2)$ such that $(s_1, s_2) \in \mathcal{R}$
Two beverage machines

\begin{align*}
\mathcal{T}_1 & \quad \text{pay} \\
\text{paid}_1 \quad \text{paid}_2 \\
\text{coke} \quad \text{soda}
\end{align*}

\begin{align*}
\mathcal{T}_2 & \quad \text{pay} \\
\text{select} \\
\text{coke} \quad \text{soda}
\end{align*}

for \( AP = \{\text{pay}, \text{coke}, \text{soda}\} \): \quad \mathcal{T}_1 \preceq \mathcal{T}_2
Two beverage machines

\[ \mathcal{T}_1 \]

\[ \mathcal{T}_2 \]

for \( \mathcal{AP} = \{ \text{pay}, \text{coke}, \text{soda} \} \): \( \mathcal{T}_1 \preceq \mathcal{T}_2 \)

simulation for \( (\mathcal{T}_1, \mathcal{T}_2) \):

\[ \{ (\text{pay, pay}), (\text{paid}_1, \text{select}), (\text{paid}_2, \text{select}), (\text{coke, coke}), (\text{soda, soda}) \} \]
Two beverage machines

For $AP = \{\text{pay}, \text{coke}, \text{soda}\}$: $T_1 \preceq T_2$, but $T_2 \not\preceq T_1$

Simulation for $(T_1, T_2)$:

$$\{ (\text{pay}, \text{pay}), (\text{paid}_1, \text{select}), (\text{paid}_2, \text{select}),
(\text{coke}, \text{coke}), (\text{soda}, \text{soda}) \}$$
Two beverage machines

\[ \mathcal{T}_1 \]

\[ \mathcal{T}_2 \]

\[
\begin{align*}
\text{for } AP &= \{\text{pay, coke, soda}\}: \quad \mathcal{T}_1 \preceq \mathcal{T}_2, \text{ but } \mathcal{T}_2 \not\preceq \mathcal{T}_1 \\
\text{for } AP &= \{\text{pay, drink}\} : 
\end{align*}
\]
Two beverage machines

\[ T_1 \xrightarrow{\text{pay}} \text{paid}_1 \xrightarrow{\text{coke}} \quad \text{paid}_1 \xrightarrow{\text{soda}} \quad \text{paid}_2 \xrightarrow{\text{coke}} \quad \text{paid}_2 \xrightarrow{\text{soda}} \]

for \( AP = \{ \text{pay, coke, soda} \} \): \( T_1 \preceq T_2 \), but \( T_2 \nsubseteq T_1 \)

for \( AP = \{ \text{pay, drink} \} \): \( T_1 \preceq T_2 \), and \( T_2 \preceq T_1 \)
Two beverage machines

for $AP = \{\text{pay, coke, soda}\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

for $AP = \{\text{pay, drink}\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, and $\mathcal{T}_2 \preceq \mathcal{T}_1$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: as before
Two beverage machines

\[
\mathcal{T}_1 \xrightarrow{\text{pay}} \mathcal{P} \xrightarrow{\text{paid}_1} \mathcal{C} \xrightarrow{\text{coke}} \mathcal{P} \xrightarrow{\text{paid}_2} \mathcal{S} \xrightarrow{\text{soda}}
\]

\[
\mathcal{T}_2 \xrightarrow{\text{pay}} \mathcal{S} \xrightarrow{\text{select}} \mathcal{C} \xrightarrow{\text{coke}} \mathcal{P} \xrightarrow{\text{paid}_1} \mathcal{P} \xrightarrow{\text{paid}_2} \mathcal{S} \xrightarrow{\text{soda}}
\]

for \( \mathcal{A} \mathcal{P} = \{\text{pay, coke, soda}\} \): \( \mathcal{T}_1 \preceq \mathcal{T}_2 \), but \( \mathcal{T}_2 \not\preceq \mathcal{T}_1 \)

for \( \mathcal{A} \mathcal{P} = \{\text{pay, drink}\} \) : \( \mathcal{T}_1 \preceq \mathcal{T}_2 \), and \( \mathcal{T}_2 \preceq \mathcal{T}_1 \)

simulation for \((\mathcal{T}_2, \mathcal{T}_1)\):
\[
\{(\text{pay, pay}), (\text{select, paid}_1), (\text{select, paid}_2), (\text{coke, coke}), (\text{soda, soda})\}
\]
Path fragment lifting for simulation $\mathcal{R}$

can be completed to
Correct or wrong?

Correct. simulation: \[ \{(s_1, s_2), (s'_1, s'_2)\} \]
Correct or wrong?

\begin{align*}
\text{correct. simulation: } & \{ (s_1, s_2), (s'_1, s'_2) \} \\
\end{align*}

\begin{align*}
\text{wrong. there is \underline{no} path fragment in } & \mathcal{T}_2 \\
\text{corresponding to the path fragment } & s_1 s'_1 s'_1
\end{align*}
Correct or wrong?
Correct or wrong?

Correct. simulation: \(\{ (s_1, s_2), (s'_1, s'_2), (s'_1, s''_2) \}\)
Correct or wrong?

correct. simulation: \{ (s_1, s_2), (s'_1, s'_2), (s'_1, s''_2) \}
Correct or wrong?

Correct.  simulation: \( \{(s_1, s_2), (s_1', s_2'), (s_1', s_2'')\} \)

wrong.  \( s_1' \not\leq s_2 \) and \( s_1' \not\leq t_2' \)
Simulation preorder ...

- as a relation that compares two transition systems
Simulation preorder ...

- as a relation that compares *two transition systems*

\[ \mathcal{T}_1 \quad \mathcal{T}_2 \]
Simulation preorder ...

- as a relation that compares two transition systems
- as a relation on the states of one transition system
Simulation preorder ...

- as a relation that compares two transition systems
- as a relation on the states of one transition system

\[ s_1 \preceq_T s_2 \] iff ?
Simulation preorder ...

- as a relation that compares two transition systems
- as a relation on the states of one transition system

\[ s_1 \preceq_T s_2 \iff \tau_{s_1} \preceq \tau_{s_2} \]

iff there exists a simulation \( R \)
for \( \tau \) with \((s_1, s_2) \in R\)
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarest relation on $S$ such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$: 


Let $\mathcal{T} = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarsest relation on $S$ such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$:

1. $L(s_1) = L(s_2)$
2. each transition of $s_1$ can be mimicked by a transition of $s_2$
Simulation preorder for a single TS

Let $T = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_T$ is the coarsest relation on $S$ such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_T s_2$:

1. $L(s_1) = L(s_2)$
2. each transition of $s_1$ can be mimicked by a transition of $s_2$

$\preceq_T$ is a preorder, i.e., transitive and reflexive.
Let $\mathcal{T}$ be a transition system with state space $S$.

A simulation for $\mathcal{T}$ is a binary relation $\mathcal{R} \subseteq S \times S$ s.t.

1. if $(s_1, s_2) \in \mathcal{R}$ then $L(s_1) = L(s_2)$

2. for all $(s_1, s_2) \in \mathcal{R}$:
   \[
   \forall s'_1 \in \text{Post}(s_1) \exists s'_2 \in \text{Post}(s_2) \text{ s.t. } (s'_1, s'_2) \in \mathcal{R}
   \]

simulation preorder $\preceq_{\mathcal{T}}$:

$s_1 \preceq_{\mathcal{T}} s_2$ iff there exists a simulation $\mathcal{R}$ for $\mathcal{T}$ s.t. $(s_1, s_2) \in \mathcal{R}$
Path fragment lifting for $\preceq_T$

can be completed to
Example: simulation preorder $\preceq_T$

$s_1 \preceq_T s_2$

Diagram:

- $s_1$ with state label $\{a\}$ connected to $s'_1$.
- $s_2$ with state label $\{a\}$ connected to $s'_2$.
- $s'_1$ connected to $s_1$.
- $s'_2$ connected to $s_2$.
Example: simulation preorder $\preceq_T$

$s_1 \preceq_T s_2$ as

$\{(s_1, s_2), (s'_1, s'_2), (s'_1, s'_1)\}$ is a simulation for $T$
Example: simulation preorder $\preceq_T$

$s_1 \preceq_T s_2$ as

$\{(s_1, s_2), (s_1', s_2'), (s_1', s_1')\}$ is a simulation for $\mathcal{T}$
Example: simulation preorder $\preceq_T$

$s_1 \preceq_T s_2$ as

$\{(s_1, s_2), (s'_1, s'_2), (s'_1, s'_1)\}$ is a simulation for $T$

$s_1 \rightarrow s'_1 \rightarrow s'_1 \rightarrow s'_1 \rightarrow \ldots$

is simulated by

$s_2 \rightarrow s'_2 \rightarrow s'_1 \rightarrow s'_1 \rightarrow \ldots$
Abstraction and simulation
Abstraction and simulation

transition system $\mathcal{T}$ with state space $\mathcal{S}$
Abstraction and simulation

transition system $\mathcal{T}$
with state space $S$

“small” abstract state space $S'$
Abstraction and simulation

transition system $\mathcal{T}$ with state space $S$

abstract transition system $\mathcal{T}_f$ with state space $S'$

abstraction function $f$

$\mathcal{T}$

$S$

$s$

$f(s)$
Abstraction and simulation

transition system $\mathcal{T}$ with state space $S$

abstraction function $f$

abstract transition system $\mathcal{T}_f$ with state space $S'$

lifting of transitions:

\[
\begin{align*}
  s & \rightarrow s' \\
  f(s) & \rightarrow f(s')
\end{align*}
\]
Abstraction and simulation

transition system $\mathcal{T}$ with state space $S$

abstract transition system $\mathcal{T}_f$ with state space $S'$

lifting of transitions:

$s \xrightarrow{s'} s' \xrightarrow{f(s)} f(s')$
given: transition system $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, \text{AP}, L)$

set $S'$ and abstraction function $f : S \rightarrow S'$

s.t. $L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$
Abstraction and simulation

given: transition system \( \mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L) \)

set \( S' \) and abstraction function \( f : S \rightarrow S' \)

s.t. \( L(s) = L(t) \) if \( f(s) = f(t) \) for all \( s, t \in S \)

goal: define abstract transition system \( \mathcal{T}_f \)

with state space \( S' \) s.t. \( \mathcal{T} \preceq \mathcal{T}_f \)
Abstraction and simulation

abstraction function \( f : S \rightarrow S' \) s.t.
\[ L(s) = L(t) \text{ if } f(s) = f(t) \text{ for all } s, t \in S \]

transition system
\[ T = (S, Act, \rightarrow, S_0, AP, L) \]
\[ \Downarrow \]
abstract transition system
\[ T_f = (S', Act', \rightarrow_f, S'_0, AP, L') \]
abstraction function \( f : S \to S' \) s.t.

\[ L(s) = L(t) \text{ if } f(s) = f(t) \text{ for all } s, t \in S \]

transition system

\[ T = (S, \text{Act}, \mathbin{\rightarrow}, S_0, \text{AP}, L) \]

\[ \Downarrow \]

abstract transition system

\[ T_f = (S', \text{Act}', \mathbin{\rightarrow}_f, S'_0, \text{AP}, L') \]

where \( S'_0 = \{ f(s_0) : s_0 \in S_0 \} \) and \( L'(f(s)) = L(s) \)

\[ s \mathbin{\rightarrow} s' \quad \frac{f(s)}{f(s) \mathbin{\rightarrow}_f f(s')} \]
Abstraction and simulation

abstraction function $f : S \rightarrow S'$ s.t.

$L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

$$
\text{transition system } \\
\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, \text{AP}, L)
$$

$$
\downarrow
$$

abstract transition system

$$
\mathcal{T}_f = (S', \text{Act}', \rightarrow_f, S'_0, \text{AP}, L')
$$

Then $\mathcal{T} \preceq \mathcal{T}_f$
Abstraction and simulation

abstraction function \( f : S \rightarrow S' \) s.t.

\[ L(s) = L(t) \text{ if } f(s) = f(t) \text{ for all } s, t \in S \]

transition system

\[ T = (S, Act, \rightarrow, S_0, AP, L) \]

abstract transition system

\[ T_f = (S', Act', \rightarrow_f, S'_0, AP, L') \]

Then \( T \preceq T_f \)

\[ \mathcal{R} = \{ \langle s, f(s) \rangle : s \in S \} \text{ is a simulation for } (T, T_f) \]
Data abstraction

\[
\text{WHILE} \ x > 0 \ \text{DO}
\]
\[
\begin{align*}
    x & := x - 1; \\
y & := y + 1
\end{align*}
\]
\text{OD}
\[
\text{IF} \ \text{even}\(y\) \ \text{THEN} \ \text{return} \ "1" \\
\text{ELSE} \ \text{return} \ "0"
\] 
\text{FI}

\[x \in \mathbb{N}\]
\[y \in \mathbb{N}\]
WHILE \( x > 0 \) DO

\[
\begin{align*}
x & := x - 1; \\
y & := y + 1
\end{align*}
\]

OD

IF \( \text{even}(y) \)

THEN return “1”

ELSE return “0”

FI

\[
\begin{align*}
x & \in \mathbb{N} & \quad \rightarrow \quad x & \in \{ \text{gzero, zero} \} \\
y & \in \mathbb{N} & \quad \rightarrow \quad y & \in \{ \text{even, odd} \}
\end{align*}
\]
Data abstraction

\[
\text{WHILE } x > 0 \text{ DO}
\begin{align*}
x &:= x - 1; \\
y &:= y + 1
\end{align*}
\text{OD}
\]

\[
\text{IF } \text{even}(y) \text{ THEN return "1"}
\text{ELSE return "0"}
\text{FI}
\]

\[
x \in \mathbb{N} \quad \rightarrow \quad x \in \{ \texttt{gzero}, \texttt{zero} \}
\]

\[
y \in \mathbb{N} \quad \rightarrow \quad y \in \{ \texttt{even}, \texttt{odd} \}
\]
WHILE $x > 0$ DO
  $x := x - 1$;
  $y := y + 1$
OD

IF $\text{even}(y)$ THEN return "1"
ELSE return "0"
FI

WHILE $x = \text{gzero}$ DO
  $x := \text{gzero}$ or $x := \text{zero}$
  IF $y = \text{even}$ THEN $y := \text{odd}$
  ELSE $y := \text{even}$
  FI
OD

IF $y = \text{even}$ THEN return "1"
ELSE return "0"
FI
While $x > 0$ do
\[
\begin{align*}
  x &:= x - 1; \\
y &:= y + 1
\end{align*}
\]
end

If $\text{even}(y)$
\[
\begin{align*}
  \text{then} & \quad \text{return } "1" \\
  \text{else} & \quad \text{return } "0"
\end{align*}
\]
end

Concrete operation:
\[
x := x - 1
\]

Abstract operation, e.g.,
\[
gzero \mapsto gzero \text{ or } zero
\]
abstract TS simulates the concrete one
WHILE $x > 0$ DO
  $x := x - 1$
  $y := y + 1$
OD
IF even($y$) THEN return 1 ELSE return 0 FI

WHILE $x = gzero$ DO
  $x := gzero$ or $x := zero$
  IF $y = even$ THEN $y := odd$ ELSE $y := even$ FI
OD
IF $y = even$ THEN return 1 ELSE return 0 FI
\( \ell_0 \) WHILE \( x > 0 \) DO
\( \ell_1 \) \( x := x - 1 \)
\( \ell_2 \) \( y := y + 1 \)
OD
\( \ell_3 \) IF \( \text{even}(y) \)
\( \ell_4 \) THEN return \( 1 \)
\( \ell_5 \) ELSE return \( 0 \)

\( \ell_0 \) WHILE \( x = \text{gzero} \) DO
\( \ell_1 \) \( x := \text{gzero} \) or \( x := \text{zero} \)
\( \ell_2 \) IF \( y = \text{even} \)
\( \quad \) THEN \( y := \text{odd} \)
\( \quad \) ELSE \( y := \text{even} \)
FI
OD
\( \ell_3 \) IF \( y = \text{even} \)
\( \ell_4 \) THEN return \( 1 \)
\( \ell_5 \) ELSE return \( 0 \) FI
\[\ell_0 \text{ WHILE } x > 0 \text{ DO}\]
\[\ell_1 x := x - 1\]
\[\ell_2 y := y + 1\]
\[\text{OD}\]
\[\ell_3 \text{ IF } \text{even}(y)\]
\[\ell_4 \text{ THEN return 1}\]
\[\ell_5 \text{ ELSE return 0}\]

\[\ell_0 \text{ WHILE } x = \text{gzero} \text{ DO}\]
\[\ell_1 x := \text{gzero or } x := \text{zero}\]
\[\ell_2 \text{ IF } y = \text{even}\]
\[\text{THEN } y := \text{odd}\]
\[\text{FI}\]
\[\text{ELSE } y := \text{even}\]
\[\text{OD}\]
\[\ell_3 \text{ IF } y = \text{even}\]
\[\ell_4 \text{ THEN return 1}\]
\[\ell_5 \text{ ELSE return 0} \text{ FI}\]
Simulation preorder vs. and trace inclusion

\[ I_1 \preceq I_2 \implies \text{Tracesfin}(I_1) \subseteq \text{Tracesfin}(I_2) \]

reason: path fragment lifting for \( \preceq \)
Simulation preorder vs. and trace inclusion

\[ \mathcal{T}_1 \preceq \mathcal{T}_2 \implies \text{Traces}_{\text{fin}}(\mathcal{T}_1) \subseteq \text{Traces}_{\text{fin}}(\mathcal{T}_2) \]

If \( \mathcal{T}_1 \) does not have terminal states, then:

\[ \mathcal{T}_1 \preceq \mathcal{T}_2 \implies \text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2) \]

... does not hold if \( \mathcal{T}_1 \) has terminal states ...

\[ \text{Traces}(\mathcal{T}_1) = \{ \emptyset \emptyset \} \neq \{ \emptyset^\omega \} = \text{Traces}(\mathcal{T}_2) \]
Simulation equivalence $\simeq_T$

kernel of the simulation preorder, i.e.,

$$\simeq = \preceq \cap \preceq^{-1}$$

For TS $\mathcal{T}_1$ and $\mathcal{T}_2$ over the same set of atomic propositions:

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \iff \mathcal{T}_1 \preceq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \preceq \mathcal{T}_1$$
Simulation equivalence $\simeq_T$

kernel of the simulation preorder, i.e.,

$$\simeq = \preceq \cap \preceq^{-1}$$

For TS $\mathcal{T}_1$ and $\mathcal{T}_2$ over the same set of atomic propositions:

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \text{ iff } \mathcal{T}_1 \preceq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \preceq \mathcal{T}_1$$

for states $s_1$ and $s_2$ of a TS $\mathcal{T}$:

$$s_1 \simeq_T s_2 \text{ iff } s_1 \preceq_T s_2 \text{ and } s_2 \preceq_T s_1$$
Two beverage machines

\( T_1: \)

- \text{pay}
- \text{soda}
- \text{coke}

\( T_2: \)

- \text{pay}
- \text{soda}
- \text{coke}
- \text{soda}'

for \( AP = \{ \text{pay}, \text{coke}, \text{soda} \} \)
Two beverage machines

$T_1$: $\text{pay} \rightarrow \text{soda} \rightarrow \text{coke} \rightarrow \text{pay}$

$T_2$: $\text{pay} \rightarrow \text{soda} \rightarrow \text{coke} \rightarrow \text{pay}$

for $AP = \{\text{pay, coke, soda}\}$

$T_2 \preceq T_1$, but $T_1 \not\equiv T_2$
Two beverage machines

\[ \mathcal{T}_1 \] :

- Pay
- \( s_1 \)
- Coke
- Soda

\[ \mathcal{T}_2 \] :

- Pay
- \( s_2 \)
- Coke
- Soda

\( \text{for } AP = \{ \text{pay}, \text{coke}, \text{soda} \} \)

\[ \mathcal{T}_2 \preceq \mathcal{T}_1, \text{ but } \mathcal{T}_1 \not\cong \mathcal{T}_2 \]

\[ \text{since } \mathcal{T}_1 \not\preceq \mathcal{T}_2 \]
Two beverage machines

$T_1$: 

![Diagram of beverage machine 1]

For $AP = \{ \text{pay, coke, soda} \}$

$T_2 \preceq T_1$, but $T_1 \nRightarrow T_2$

since $T_1 \nRightarrow T_2$

$T_2$: 

![Diagram of beverage machine 2]

For $AP = \{ \text{pay, drink} \}$:
Two beverage machines

\( \mathcal{T}_1: \)

\[
\begin{array}{c}
\text{pay} \\
\downarrow \\
\text{soda} \\
\downarrow \\
\text{coke} \\
\end{array}
\]

\( \mathcal{T}_2: \)

\[
\begin{array}{c}
\text{pay} \\
\downarrow \\
\text{soda} \\
\downarrow \\
\text{coke} \\
\end{array}
\]

for \( AP = \{ \text{pay, coke, soda} \} \)

\( \mathcal{T}_2 \preceq \mathcal{T}_1, \text{ but } \mathcal{T}_1 \not\equiv \mathcal{T}_2 \quad \text{ since } \mathcal{T}_1 \not\equiv \mathcal{T}_2 \)

for \( AP = \{ \text{pay, drink} \} : \quad \mathcal{T}_1 \simeq \mathcal{T}_2 \)
Example: simulation equivalent TS

$\mathcal{T}_1$: 

$\mathcal{T}_2$: 

$T_1$:$s_1$ 

$t_1$ 

$u_1$ 

$T_2$: 

$s_2$ 

$t_3$ 

$t_2$ 

$u_2$
Example: simulation equivalent TS

\[ \mathcal{T}_1: \]
\[ \mathcal{T}_2: \]

simulation for \((\mathcal{T}_1, \mathcal{T}_2)\):

\[ \{(s_1, s_2), (t_1, t_2), (u_1, u_2)\} \]
Example: simulation equivalent TS

\[ T_1: \]

\[ T_2: \]

\[
\text{simulation for } (T_1, T_2): \\
\{ (s_1, s_2), (t_1, t_2), (u_1, u_2) \}
\]

\[
\text{simulation for } (T_2, T_1): \\
\{ (s_2, s_1), (t_2, t_1), (t_3, t_1), (u_2, u_1) \}
\]
Bisimulation vs. simulation equivalence

Bisimulation equivalence \( \sim \) is strictly finer than simulation equivalence \( \simeq \)
Bisimulation vs. simulation equivalence

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

That is:

1. $\mathcal{I}_1 \sim \mathcal{I}_2$ implies $\mathcal{I}_1 \simeq \mathcal{I}_2$

   \textit{Proof:} Let $\mathcal{R}$ is a bisimulation for $(\mathcal{I}_1, \mathcal{I}_2)$.
   
   \begin{itemize}
     \item $\mathcal{R}$ is a simulation for $(\mathcal{I}_1, \mathcal{I}_2) \implies \mathcal{I}_1 \leq \mathcal{I}_2$
     \item $\mathcal{R}^{-1}$ is a simulation for $(\mathcal{I}_2, \mathcal{I}_1) \implies \mathcal{I}_2 \leq \mathcal{I}_1$
   \end{itemize}

2. there exist TS $\mathcal{I}_1$ and $\mathcal{I}_2$ s.t. $\mathcal{I}_1 \simeq \mathcal{I}_2$ and $\mathcal{I}_1 \not\sim \mathcal{I}_2$
bisimulation equivalence

\[ s_1 \sim s_2 \]

\[ s'_1 \]
bisimulation equivalence

\[ s_1 \sim s_2 \]

\[ s_1' \sim s_2' \]
bisimulation equivalence

\[
\begin{array}{c}
S_1 \sim S_2 \\
S'_1 \sim S'_2
\end{array}
\]

simulation equivalence

\[
\begin{array}{c}
S_1 \sim S_2 \\
S'_1
\end{array}
\]
bisimulation equivalence

\[s_1 \sim s_2\]

\[s'_1 \sim s'_2\]

simulation equivalence

\[s_1 \sim s_2\]

\[s'_1 \preceq s'_2\]
bisimulation equivalence

\[ S_1 \sim S_2 \]

\[ S'_1 \sim S'_2 \]

simulation equivalence

\[ S_1 \sim S_2 \]

\[ S'_1 \preceq S'_2 \]
bisimulation equivalence

\[ s_1 \sim s_2 \]

\[ s_1' \sim s_2' \]

simulation equivalence

\[ s_1 \sim s_2 \]

\[ s_1' \preceq s_2' \]

\[ T_1 \]

\[ T_2 \]
bisimulation equivalence

\[ s_1 \sim s_2 \]

\[ s_1' \sim s_2' \]

simulation equivalence

\[ s_1 \sim s_2 \]

\[ s_1' \preceq s_2' \]

\[ \mathcal{T}_2 \preceq \mathcal{T}_1 \text{, as } \mathcal{T}_2 \text{ is a "subsystem" of } \mathcal{T}_1 \]
bisimulation equivalence

$$s_1 \sim s_2$$

$$s_1' \sim s_2'$$

simulation equivalence

$$s_1 \sim s_2$$

$$s_1' \preceq s_2'$$

simulation for $$(\mathcal{T}_1, \mathcal{T}_2)$$:

$$\{(s_1, s_2), (s'_1, s'_2), (s''_1, s'_2), (u_1, u_2), (v_1, v_2)\}$$
Simulation vs trace equivalence

\[ T_1 \simeq T_2 \iff \text{Traces}(T_1) = \text{Traces}(T_2) \]

\[ \text{Traces}(T_1) = \text{Traces}(T_2) \implies T_1 \simeq T_2 \]

not trace equivalent but simulation equivalent

not trace equivalent not simulation equivalent
Simulation vs trace equivalence

\[ T_1 \sim T_2 \nleftrightarrow Traces(T_1) = Traces(T_2) \]

\[ Traces(T_1) = Traces(T_2) \nleftrightarrow T_1 \sim T_2 \]

\[ \begin{align*}
&\text{not trace equivalent} \\
&\text{but simulation equivalent}
\end{align*} \]

\[ \begin{align*}
&\text{trace equivalent} \\
&\text{not simulation equivalent}
\end{align*} \]
Simulation vs. finite trace equivalence

\[ T_1 \sim T_2 \iff \text{Traces}(T_1) = \text{Traces}(T_2) \]

\[ \text{Traces}(T_1) = \text{Traces}(T_2) \iff T_1 \sim T_2 \]

\[ T_1 \sim T_2 \implies \text{Tracesfin}(T_1) = \text{Tracesfin}(T_2) \]

while "\(\iff\)" does not hold
Simulation vs. finite trace equivalence

\[ T_1 \simeq T_2 \quad \not\Rightarrow \quad \text{Traces}(T_1) = \text{Traces}(T_2) \]

\[ \text{Traces}(T_1) = \text{Traces}(T_2) \quad \not\Rightarrow \quad T_1 \simeq T_2 \]

\[ T_1 \simeq T_2 \quad \Rightarrow \quad \text{Tracesfin}(T_1) = \text{Tracesfin}(T_2) \]

while "\(\Leftarrow\)" does not hold

If \( T_1, T_2 \) do not have terminal states then:

\[ T_1 \simeq T_2 \quad \Rightarrow \quad \text{Traces}(T_1) = \text{Traces}(T_2) \]
Summary: trace and (bi)simulation relations
bisimulation equivalence
\[ T_1 \sim T_2 \]

simulation equivalence
\[ T_1 \simeq T_2 \]

simulation preorder
\[ T_1 \preceq T_2 \]
bisimulation equivalence \( \mathcal{T}_1 \sim \mathcal{T}_2 \)

simulation equivalence \( \mathcal{T}_1 \simeq \mathcal{T}_2 \)

finite trace equivalence \( \text{Traces}_{\text{fin}}(\mathcal{T}_1) = \text{Traces}_{\text{fin}}(\mathcal{T}_2) \)

simulation preorder \( \mathcal{T}_1 \preceq \mathcal{T}_2 \)

finite trace inclusion \( \text{Traces}_{\text{fin}}(\mathcal{T}_1) \subseteq \text{Traces}_{\text{fin}}(\mathcal{T}_2) \)
bisimulation equivalence $\mathcal{T}_1 \sim \mathcal{T}_2$

simulation equivalence $\mathcal{T}_1 \simeq \mathcal{T}_2$

finite trace equivalence $\text{Traces}_{\text{fin}}(\mathcal{T}_1) = \text{Traces}_{\text{fin}}(\mathcal{T}_2)$

finite trace inclusion $\text{Traces}_{\text{fin}}(\mathcal{T}_1) \subseteq \text{Traces}_{\text{fin}}(\mathcal{T}_2)$

simulation preorder $\mathcal{T}_1 \preceq \mathcal{T}_2$

trace equivalence $\text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2)$

trace inclusion $\text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2)$
bisimulation equivalence
\[ \mathcal{T}_1 \sim \mathcal{T}_2 \]

finite trace equivalence
\[ \text{Traces}_{\text{fin}}(\mathcal{T}_1) = \text{Traces}_{\text{fin}}(\mathcal{T}_2) \]

trace inclusion
\[ \text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2) \]

without terminal states

simulation preorder
\[ \mathcal{T}_1 \preceq \mathcal{T}_2 \]

finite trace inclusion
\[ \text{Traces}_{\text{fin}}(\mathcal{T}_1) \subseteq \text{Traces}_{\text{fin}}(\mathcal{T}_2) \]

simulation equivalence
\[ \mathcal{T}_1 \simeq \mathcal{T}_2 \]

trace equivalence
\[ \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \]
bisimulation equivalence
\[ T_1 \sim T_2 \]

finite trace equivalence
\[ \text{Traces}_{\text{fin}}(T_1) = \text{Traces}_{\text{fin}}(T_2) \]

trace inclusion
\[ \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \]

without terminal states

simulation preorder
\[ T_1 \preceq T_2 \]

simulation equivalence
\[ T_1 \simeq T_2 \]

trace equivalence
\[ \text{Traces}(T_1) = \text{Traces}(T_2) \]

finite trace inclusion
\[ \text{Traces}_{\text{fin}}(T_1) \subseteq \text{Traces}_{\text{fin}}(T_2) \]

AP-determinism

without terminal states
Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS.

$\mathcal{T}$ is called $AP$-deterministic iff

1. for all states $s$ and all subsets $A$ of $AP$:
   $$\left| \left\{ t \in S : s \rightarrow t \land L(t) = A \right\} \right| \leq 1$$

2. for all subsets $A$ of $AP$:
   $$\left| \left\{ s_0 \in S_0 : L(s_0) = A \right\} \right| \leq 1$$
Let $T$ be $AP$-deterministic and $s_1$, $s_2$ states in $T$.

If $\text{Traces}_{\text{fin}}(s_1) = \text{Traces}_{\text{fin}}(s_2)$ then

$$\text{Traces}(s_1) = \text{Traces}(s_2)$$

mainly because:

- each (finite or infinite) word $\sigma_1$ over $2^{AP}$ is induced by at most one path fragment starting in $s_1$ or $s_2$, respectively

- if $\sigma = A_0A_1 \ldots A_iA_{i+1} \ldots \in \text{Traces}(s_1)$ then there is no proper prefix $A_0A_1 \ldots A_i$ of $\sigma$ belongs to $\text{Traces}(s_1)$

+ analogous statement for $s_2$
Let $\mathcal{T}$ be $AP$-deterministic and $s_1$, $s_2$ states in $\mathcal{T}$.

If $\text{Traces}_{\text{fin}}(s_1) \subseteq \text{Traces}_{\text{fin}}(s_2)$ then

$\text{Traces}(s_1) \subseteq \text{Traces}(s_2)$
Let $\mathcal{T}$ be $AP$-deterministic and $s_1$, $s_2$ states in $\mathcal{T}$.

If $Traces_{\text{fin}}(s_1) \subseteq Traces_{\text{fin}}(s_2)$ then

$Traces(s_1) \subseteq Traces(s_2)$

wrong.

$Traces_{\text{fin}}(s_1) \subseteq Traces_{\text{fin}}(s_2)$

$\bullet \in Traces(s_1) \setminus Traces(s_2)$
(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be $AP$-deterministic and $s_1$, $s_2$ states in $\mathcal{T}$.
Then the following statements are equivalent:

1. $s_1 \sim_{\mathcal{T}} s_2$ (bisimulation equivalence)
2. $s_1 \simeq_{\mathcal{T}} s_2$ (simulation equivalence)
3. $\text{Traces}_{\text{fin}}(s_1) = \text{Traces}_{\text{fin}}(s_2)$
4. $\text{Traces}(s_1) = \text{Traces}(s_2)$

(1) $\implies$ (2): $\checkmark$
(2) $\implies$ (3): ... path fragment lifting ...
(3) $\implies$ (4): just shown
(4) $\implies$ (1): ...
Bisimulation and trace equivalence

Let $\mathcal{T}$ be $AP$-deterministic and $s_1, s_2$ states in $\mathcal{T}$. Then:

$$Traces(s_1) = Traces(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$
Let $\mathcal{T}$ be $AP$-deterministic and $s_1$, $s_2$ states in $\mathcal{T}$. Then:

$$\text{Traces}(s_1) = \text{Traces}(s_2)$$ implies $s_1 \sim_T s_2$

*Proof:* show that

$$\mathcal{R} = \{(s_1, s_2) : \text{Traces}(s_1) = \text{Traces}(s_2)\}$$

is a bisimulation.
Bisimulation and trace equivalence

Let $T$ be AP-deterministic and $s_1, s_2$ states in $T$. Then:

$$\text{Traces}(s_1) = \text{Traces}(s_2) \text{ implies } s_1 \sim_T s_2$$

**Proof:** show that

$$\mathcal{R} = \{(s_1, s_2) : \text{Traces}(s_1) = \text{Traces}(s_2)\}$$

is a bisimulation.

Note that if $s \rightarrow t$ then
Bisimulation and trace equivalence

Let \( T \) be \( AP \)-deterministic and \( s_1, s_2 \) states in \( T \). Then:

\[
\text{Traces}(s_1) = \text{Traces}(s_2) \implies s_1 \sim_T s_2
\]

Proof: show that

\[
\mathcal{R} = \{ (s_1, s_2) : \text{Traces}(s_1) = \text{Traces}(s_2) \}
\]

is a bisimulation.

Note that if \( s \rightarrow t \) then

\[
\text{Traces}(t) = \{ L(t)B_1B_2B_3 \ldots \in (2^{AP})^+ \cup (2^{AP})^\omega : L(s)L(t)B_1B_2B_3 \ldots \in \text{Traces}(s) \}
\]
Let $\mathcal{T}$ be $AP$-deterministic and $s_1$, $s_2$ states in $\mathcal{T}$. Then:

$$Traces_{\text{fin}}(s_1) = Traces_{\text{fin}}(s_2) \implies s_1 \sim_{\mathcal{T}} s_2$$
Let $T$ be $AP$-deterministic and $s_1, s_2$ states in $T$. Then:

$$\text{Traces}_{\text{fin}}(s_1) = \text{Traces}_{\text{fin}}(s_2) \implies s_1 \sim_T s_2$$

**Proof:** show that

$$\mathcal{R} = \{(s_1, s_2) : \text{Traces}_{\text{fin}}(s_1) = \text{Traces}_{\text{fin}}(s_2)\}$$

is a bisimulation.

Note that if $s \rightarrow t$ then

$$\text{Traces}_{\text{fin}}(t) = \{ L(t)B_1B_2\ldots B_n \in (2^{AP})^+ :$$

$$L(s)L(t)B_1B_2\ldots B_n \in \text{Traces}_{\text{fin}}(s) \}$$
Trace and (bi)simulation equivalence

\[ \mathcal{T}_1 \sim \mathcal{T}_2 \]

Simulation equivalence

\[ \mathcal{T}_1 \simeq \mathcal{T}_2 \]

Trace equivalence

\[ \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \]

Finite trace equivalence

\[ \text{Traces}_{\text{fin}}(\mathcal{T}_1) = \text{Traces}_{\text{fin}}(\mathcal{T}_2) \]
For AP-deterministic TS

- Bisimulation equivalence: $\mathcal{T}_1 \sim \mathcal{T}_2$
- Simulation equivalence: $\mathcal{T}_1 \simeq \mathcal{T}_2$
- Trace equivalence: $\text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2)$
- Finite trace equivalence: $\text{Traces}_{\text{fin}}(\mathcal{T}_1) = \text{Traces}_{\text{fin}}(\mathcal{T}_2)$
For AP-deterministic TS

\[ \mathcal{I}_1 \sim \mathcal{I}_2 \]

\[ \text{trace equivalence} \quad \text{Traces}(\mathcal{I}_1) = \text{Traces}(\mathcal{I}_2) \]

\[ \text{finite trace equivalence} \quad \text{Traces}_{\text{fin}}(\mathcal{I}_1) = \text{Traces}_{\text{fin}}(\mathcal{I}_2) \]

\[ \text{simulation equivalence} \quad \mathcal{I}_1 \simeq \mathcal{I}_2 \]

\[ \text{AP-determinism} \]
Logical characterizations

\(\text{LT safety prop.}\)

\(\text{trace inclusion}\)

\(\text{trace inclusion}\)

\(\text{trace equivalence}\)

\(\text{bisimulation equivalence} \sim\)

\(\text{LTL}\)

\(\text{LTL}\)

\(\text{LTL}^*\)

\(\text{CTL}\)
Logical characterizations

**LTL**
- safety prop.
- finite trace inclusion
- trace inclusion
- trace equivalence
- stutter trace equivalence

**LTL\_O**
- stutter trace equivalence

**LTL**
- bisimulation equivalence
- stutter bis. equiv. with div.

**CTL**
- \(\Delta\)

**CTL\***
- \(\equiv\)
Logical characterizations

- **LT** safety prop.
- **LTL**
  - finite trace inclusion
  - trace inclusion
  - trace equivalence
  - stutter trace equivalence
  
- **LTL**
  - simulation preorder $\preceq$
  - bisimulation equivalence $\sim$
  - stutter bis. equiv. with div. $\approx_{\text{div}}$

- **CTL**
  - $\bullet$

- **CTL***
  - $\bullet$

For TS without terminal states
Logical characterizations

\[ \text{LT safety prop.} \]

\[ \text{LTL} \]

\[ \text{LTL} \]

\[ \text{LTL} \]

\[ \forall \text{CTL*} \]

\[ \text{CTL*} \]

\[ \text{CTL*} \]

\[ \text{CTL} \]

\[ \text{CTL} \]

finite trace inclusion

trace inclusion

trace equivalence

bisimulation equivalence

simulation preorder \( \preceq \)

for TS without terminal states

\[ \forall \text{CTL*} \]

\[ \text{CTL*} \]

\[ \text{CTL} \]

\[ \text{CTL} \]

stutter trace equivalence \( \triangleq \)

stutter bis. equiv. with div. \( \approx^{\text{div}} \)

LTL

\[ \text{LTL} \]

\[ \text{LTL} \]

\[ \text{LTL} \]
for bisimulation equivalence $\sim_T$:

\[
\begin{align*}
  s_1 \sim_T s_2 &\text{ iff } s_1, s_2 \text{ satisfy the same } \mathbf{CTL}^* \text{ formulas} \\
  &\text{iff } s_1, s_2 \text{ satisfy the same } \mathbf{CTL} \text{ formulas}
\end{align*}
\]
for bisimulation equivalence $\sim_T$:

$$s_1 \sim_T s_2 \iff s_1, s_2 \text{ satisfy the same } \text{CTL}^* \text{ formulas}$$

iff $s_1, s_2$ satisfy the same $\text{CTL}$ formulas

for the simulation preorder $\preceq_T$:

by a sublogic $L$ of $\text{CTL}^*$ that subsumes $\text{LTL}$

$$s_1 \preceq_T s_2 \iff \text{for all formulas } \Phi \in L: \quad s_2 \models \Phi \text{ implies } s_1 \models \Phi$$

observation: $L$ cannot be closed under negation
The universal fragment $\forall \text{CTL}^*$ of $\text{CTL}^*$

$\text{CTL}^*$ formulas in positive normal form, without $\exists$
Syntax of ∀CTL*

∀CTL* state formulas:

\[ \Phi ::= \text{true} \mid \text{false} \mid a \mid \neg a \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid \forall \varphi \]

∀CTL* path formulas:

\[ \varphi ::= \Phi \mid \varphi_1 \land \varphi_2 \mid \varphi_1 \lor \varphi_2 \mid \O \varphi \mid \varphi_1 \cup \varphi_2 \mid \varphi_1 \mathsf{W} \varphi_2 \]
Syntax of $\forall$CTL*

$\forall$CTL* state formulas:

$$\Phi ::= \text{true} \mid \text{false} \mid a \mid \neg a \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid \forall \varphi$$

$\forall$CTL* path formulas:

$$\varphi ::= \Phi \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid \bigcirc \varphi \mid \Phi_1 \mathsf{U} \Phi_2 \mid \Phi_1 \mathsf{W} \Phi_2$$

evitably: $\Diamond \varphi \overset{\text{def}}{=} \text{true} \mathsf{U} \varphi$

always: $\Box \varphi \overset{\text{def}}{=} \varphi \mathsf{W} \text{false}$
Embedding of LTL in ∀CTL*

∀CTL* state formulas:

\[ \Phi ::= true | false | a | \neg a | \Phi_1 \land \Phi_2 | \Phi_1 \lor \Phi_2 | \forall \varphi \]

∀CTL* path formulas:

\[ \varphi ::= \Phi | \varphi_1 \land \varphi_2 | \varphi_1 \lor \varphi_2 | \bigcirc \varphi | \varphi_1 \mathcal{U} \varphi_2 | \varphi_1 \mathcal{W} \varphi_2 \]

for all LTL formulas \( \varphi \) in PNF:

\[ s \models_{\text{LTL}} \varphi \text{ iff } s \models_{\forall \text{CTL}*} \forall \varphi \]

but \( \forall \diamond \forall \square a \) cannot be expressed in LTL
The universal fragments of CTL* and CTL

**syntax of ∀CTL***:

\[ \Phi ::= \text{true} \mid \text{false} \mid a \mid \neg a \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid \forall \varphi \]

\[ \varphi ::= \Phi \mid \varphi_1 \land \varphi_2 \mid \varphi_1 \lor \varphi_2 \mid \neg \varphi \mid \varphi_1 \mathbf{U} \varphi_2 \mid \varphi_1 \mathbf{W} \varphi_2 \]

∀CTL: sublogic of ∀CTL*

- no Boolean operators for paths formulas
- the arguments of the temporal modalities \( \mathbf{U} \), \( \mathbf{U} \) and \( \mathbf{W} \) are state formulas
The universal fragments of $\forall$CTL* and CTL

**Syntax of $\forall$CTL*:**

$$\Phi ::= \text{true} \mid \text{false} \mid a \mid \neg a \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid \forall \varphi$$

$$\varphi ::= \Phi \mid \varphi_1 \land \varphi_2 \mid \varphi_1 \lor \varphi_2 \mid \Box \varphi \mid \varphi_1 \mathcal{U} \varphi_2 \mid \varphi_1 \mathcal{W} \varphi_2$$

$\forall$CTL: sublogic of $\forall$CTL*

**Syntax of $\forall$CTL:**

$$\Phi ::= \text{true} \mid \text{false} \mid a \mid \neg a \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid$$

$$\forall \Box \Phi \mid \forall(\Phi_1 \mathcal{U} \Phi_2) \mid \forall(\Phi_1 \mathcal{W} \Phi_2)$$
Let $\mathcal{T}$ be a finite TS without terminal states. Then, for all states $s_1$ and $s_2$ in $\mathcal{T}$, the following statements are equivalent:

1. $s_1 \preceq_T s_2$
2. For all $\forall$CTL state formulas $\Phi$: if $s_2 \models \Phi$ then $s_1 \models \Phi$
3. For all $\forall$CTL* state formulas $\Phi$: if $s_2 \models \Phi$ then $s_1 \models \Phi$
∀CTL and simulation

\[ T_1: \]
\[ \emptyset \xrightarrow{a} \{a\} \]
\[ \emptyset \xrightarrow{a} \{a\} \]

\[ T_2: \]
\[ \emptyset \xrightarrow{a} \{a\} \]
\[ \emptyset \xrightarrow{a} \{a\} \]

\[ AP = \{a\} \]
∀CTL and simulation

\( \mathcal{T}_1: \)

\( \emptyset \)

\( \{a\} \)

\( \{a\} \)

\( \emptyset \)

\( \{a\} \)

\( \mathcal{T}_2: \)

\( \emptyset \)

\( \{a\} \)

\( \{a\} \)

AP = \( \{a\} \)
∀CTL and simulation

\[ \mathcal{T}_1 : \quad \emptyset \xrightarrow{a} \{a\} \xrightarrow{a} \{a\} \xrightarrow{a} \emptyset \]
\[ \mathcal{T}_2 : \quad \emptyset \xrightarrow{a} \{a\} \xrightarrow{a} \{a\} \xrightarrow{a} \emptyset \]

\[ AP = \{a\} \]

e.g.,

\[ \mathcal{T}_1 \not\models \forall \bigodot (\forall \bigodot \neg a \lor \forall \bigodot a) \]
\[ \mathcal{T}_2 \models \forall \bigodot (\forall \bigodot \neg a \lor \forall \bigodot a) \]
\[ \mathcal{T}_1 \not\models \forall \bigodot (\forall \square \neg a \lor \forall \square a) \]
\[ \mathcal{T}_2 \models \forall \bigodot (\forall \square \neg a \lor \forall \square a) \]
∀CTL/∀CTL* and the simulation preorder

For finite TS without terminal states, the following statements are equivalent:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>$s_1 \preceq_T s_2$</td>
<td></td>
</tr>
<tr>
<td>(2)</td>
<td>for all ∀CTL formulas $\Phi$: $s_2 \models \Phi$ implies $s_1 \models \Phi$</td>
<td></td>
</tr>
<tr>
<td>(3)</td>
<td>for all ∀CTL* formulas $\Phi$: $s_2 \models \Phi$ implies $s_1 \models \Phi$</td>
<td></td>
</tr>
</tbody>
</table>
∀CTL/∀CTL* and the simulation preorder

For finite TS without terminal states, the following statements are equivalent:

(1) $s_1 \preceq_T s_2$

(2) for all $∀CTL$ formulas $\Phi$: $s_2 \models \Phi$ implies $s_1 \models \Phi$

(3) for all $∀CTL^*$ formulas $\Phi$: $s_2 \models \Phi$ implies $s_1 \models \Phi$

(3) $\Rightarrow$ (2): obvious as $∀CTL$ is a sublogic of $∀CTL^*$
∀CTL/∀CTL* and the simulation preorder

For finite TS without terminal states, the following statements are equivalent:

<table>
<thead>
<tr>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) ( s_1 \preceq_T s_2 )</td>
</tr>
<tr>
<td>(2) for all ( \forall \text{CTL} ) formulas ( \Phi ): ( s_2 \models \Phi ) implies ( s_1 \models \Phi )</td>
</tr>
<tr>
<td>(3) for all ( \forall \text{CTL}* ) formulas ( \Phi ): ( s_2 \models \Phi ) implies ( s_1 \models \Phi )</td>
</tr>
</tbody>
</table>

(3) \( \Rightarrow \) (2): obvious as \( \forall \text{CTL} \) is a sublogic of \( \forall \text{CTL}* \)

(1) \( \Rightarrow \) (3): holds for arbitrary (possibly infinite) TS without terminal states

\[ \text{proof by structural induction} \]
∀CTL/∀CTL* and the simulation preorder

For finite TS without terminal states, the following statements are equivalent:

(1) \( s_1 \preceq_T s_2 \)

(2) for all ∀CTL formulas \( \Phi: s_2 \models \Phi \) implies \( s_1 \models \Phi \)

(3) for all ∀CTL* formulas \( \Phi: s_2 \models \Phi \) implies \( s_1 \models \Phi \)

(1) \( \implies \) (3): show by structural induction:

(i) for all ∀CTL* state formulas \( \Phi \) and states \( s_1, s_2 \):
    if \( s_1 \preceq_T s_2 \) and \( s_2 \models \Phi \) then \( s_1 \models \Phi \)

(ii) for all ∀CTL* path formulas \( \varphi \) and paths \( \pi_1, \pi_2 \):
    if \( \pi_1 \preceq_T \pi_2 \) and \( \pi_2 \models \varphi \) then \( \pi_1 \models \varphi \)
\(\forall CTL/\forall CTL^*\) and the simulation preorder

For finite TS without terminal states, the following statements are equivalent:

1. \(s_1 \preceq_T s_2\)
2. For all \(\forall CTL\) formulas \(\Phi\): \(s_2 \models \Phi\) implies \(s_1 \models \Phi\)
3. For all \(\forall CTL^*\) formulas \(\Phi\): \(s_2 \models \Phi\) implies \(s_1 \models \Phi\)

\((2) \implies (1):\) show that for finite TS:

\[\mathcal{R} = \{(s_1, s_2) : \text{ for all } \forall CTL \text{ formulas } \Phi: s_2 \models \Phi \text{ implies } s_1 \models \Phi\}\]

is a simulation.
Duality of $\forall\text{CTL}^*$ and $\exists\text{CTL}^*$

$\exists\text{CTL}^*$ (state) formulas:

$$\Psi ::= \text{true} \mid \text{false} \mid a \mid \neg a \mid \Psi_1 \land \Psi_2 \mid \Psi_1 \lor \Psi_2 \mid \exists \phi$$

$\exists\text{CTL}^*$ path formulas:

$$\phi ::= \Psi \mid \phi_1 \land \phi_2 \mid \phi_1 \lor \phi_2 \mid \Box \phi \mid \phi_1 \mathcal{U} \phi_2 \mid \phi_1 \mathcal{W} \phi_2$$

analogous: $\exists\text{CTL}$

For each $\forall\text{CTL}^*$ formula $\Phi$ there is a $\exists\text{CTL}^*$ formula $\Psi$ s.t. $\Phi \equiv \neg \Psi$ (and vice versa)

For each $\forall\text{CTL}$ formula $\Phi$ there is a $\exists\text{CTL}$ formula $\Psi$ s.t. $\Phi \equiv \neg \Psi$ (and vice versa)
If $s_1$ and $s_2$ are states in a finite TS then the following statements are equivalent:

(1) $s_1 \preceq_T s_2$

(2) for all $\forall \text{CTL}$ formulas $\Phi$:
    
    \[
    \text{if } s_2 \models \Phi \text{ then } s_1 \models \Phi
    \]

(3) for all $\forall \text{CTL}^*$ formulas $\Phi$:
    
    \[
    \text{if } s_2 \models \Phi \text{ then } s_1 \models \Phi
    \]
If $s_1$ and $s_2$ are states in a finite TS then the following statements are equivalent:

(1) $s_1 \leq_T s_2$

(2∀) for all ∀CTL formulas $\Phi$:
\[
\text{if } s_2 \models \Phi \text{ then } s_1 \models \Phi
\]

(3∀) for all ∀CTL* formulas $\Phi$:
\[
\text{if } s_2 \models \Phi \text{ then } s_1 \models \Phi
\]

(2∃) for all ∃CTL formulas $\Psi$:
\[
\text{if } s_1 \models \Psi \text{ then } s_2 \models \Psi
\]

(3∃) for all ∃CTL formulas $\Psi$:
\[
\text{if } s_1 \models \Psi \text{ then } s_2 \models \Psi
\]
Example: $\forall\text{CTL}/\exists\text{CTL}$ and simulation

$T_1$: $\emptyset \xrightarrow{a} \{a\}$

$T_2$: $\emptyset \xrightarrow{a} \{a\}$

$T_1 \not\preceq T_2$
Example: ∀CTL/∃CTL and simulation

$$\mathcal{T}_1:$$

$$\mathcal{T}_2:$$

$$\forall CTL$$ formula

$$\exists CTL$$ formula
Characterizations of simulation equivalence

for finite TS without terminal states:

\[ \mathcal{T}_1 \sim \mathcal{T}_2 \quad \text{iff} \quad \mathcal{T}_1 \preceq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \preceq \mathcal{T}_1 \]

iff \( \mathcal{T}_1, \mathcal{T}_2 \) satisfy the same \( \forall \text{CTL}^* \) formulas

iff \( \mathcal{T}_1, \mathcal{T}_2 \) satisfy the same \( \forall \text{CTL} \) formulas

iff \( \mathcal{T}_1, \mathcal{T}_2 \) satisfy the same \( \exists \text{CTL}^* \) formulas

iff \( \mathcal{T}_1, \mathcal{T}_2 \) satisfy the same \( \exists \text{CTL} \) formulas

... even holds for \( \forall \text{CTL}^* \setminus \{u,w\}, \forall \text{CTL} \setminus \{u,w\}, \exists \text{CTL}^* \setminus \{u,w\}, \exists \text{CTL} \setminus \{u,w\} \)
Simulation equivalence

\[ T_1 \]

\[ T_2 \]

\[ \hat{=} \{ a \} \]

\[ \hat{=} \{ b \} \]
Simulation equivalence

\(\mathcal{T}_1, \mathcal{T}_2\) cannot be distinguished by the temporal logics
\(\forall\text{CTL}, \forall\text{CTL}^*, \exists\text{CTL},\) or \(\exists\text{CTL}^*,\)
Simulation equivalence

\[ T_1 \cong T_2 \]

\[ \not\equiv T_1 \not\equiv T_2 \]

\[ \hat{T} = \{ a \} \]
\[ \hat{T} = \{ b \} \]

\( T_1, T_2 \) cannot be distinguished by the temporal logics \( \forall \text{CTL}, \forall \text{CTL}^*, \exists \text{CTL}, \) or \( \exists \text{CTL}^* \),

but by \( \text{CTL} \):

\[ T_1 \not\models \forall \bigcirc (\exists \bigcirc a \land \exists \bigcirc b) \]
\[ T_2 \models \forall \bigcirc (\exists \bigcirc a \land \exists \bigcirc b) \]
Does there exist a $\exists \mathcal{CTL}$ formula $\Phi$ s.t. $
abla_1 \models \Phi$ and $\nabla_2 \not\models \Phi$?
Does there exist ...?

Does there exist a $\exists\text{CTL}$ formula $\Phi$ s.t. $T_1 \models \Phi$ and $T_2 \not\models \Phi$?

Yes, as $T_1 \not\preceq T_2$, e.g., $\Phi = \exists \Diamond (\exists \Diamond a \land \exists \Diamond b)$
Does there exist a $\exists\text{CTL}$ formula $\Phi$ s.t. $T_1 \models \Phi$ and $T_2 \not\models \Phi$?

Yes, as $T_1 \not\preceq T_2$, e.g., $\Phi = \exists \bigcirc (\exists \bigcirc a \land \exists \bigcirc b)$

Does there exist a $\forall\text{CTL}$ formula $\Phi$ s.t. $T_1 \models \Phi$ and $T_2 \not\models \Phi$?
Does there exist ...?

Does there exist a $\exists \text{CTL}$ formula $\Phi$ s.t.

$\mathcal{T}_1 \models \Phi$ and $\mathcal{T}_2 \not\models \Phi$?

Yes, as $\mathcal{T}_1 \not\preceq \mathcal{T}_2$, e.g., $\Phi = \exists O (\exists O a \land \exists O b)$

Does there exist a $\forall \text{CTL}$ formula $\Phi$ s.t.

$\mathcal{T}_1 \models \Phi$ and $\mathcal{T}_2 \not\models \Phi$?

No, as $\mathcal{T}_2 \preceq \mathcal{T}_1$
Does there exist a $\exists\text{CTL}$ formula $\Phi$ s.t.

$\mathcal{T}_1 \models \Phi$ and $\mathcal{T}_2 \not\models \Phi$?
Does there exist a $\exists$CTL formula $\Phi$ s.t.

\[ T_1 \models \Phi \text{ and } T_2 \not\models \Phi \]?

No, since $T_1 \simeq T_2$
Does there exist a $\exists\text{CTL}$ formula $\Phi$ s.t. $\mathcal{T}_1 \models \Phi$ and $\mathcal{T}_2 \not\models \Phi$?

No, since $\mathcal{T}_1 \simeq \mathcal{T}_2$

Simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: $\{ (s_1, s_2), (v_1, s_2), (t_1, t_2) \}$
Does there exist \( \exists \mathcal{CTL} \) formula \( \Phi \) s.t.

\[
\mathcal{T}_1 \models \Phi \quad \text{and} \quad \mathcal{T}_2 \not\models \Phi
\]

\textbf{no, since } \mathcal{T}_1 \simeq \mathcal{T}_2

simulation for \((\mathcal{T}_1, \mathcal{T}_2)\): \(\{(s_1, s_2), (v_1, s_2), (t_1, t_2)\}\)

simulation for \((\mathcal{T}_2, \mathcal{T}_1)\):

\(\{(s_2, s_1), (s_2, v_1), (v_2, v_1), (t_1, t_2)\}\)
Does there exist a $\textbf{CTL}$ formula $\Phi$ s.t. $\mathcal{T}_1 \not\models \Phi$ and $\mathcal{T}_2 \models \Phi$?
Does there exist a \(\mathbf{CTL}\) formula \(\Phi\) s.t. \(\mathcal{T}_1 \not\models \Phi\) and \(\mathcal{T}_2 \models \Phi\)?

**Yes**, as \(\mathcal{T}_1 \not\sim \mathcal{T}_2\), e.g., \(\Phi = \exists \bigodot \forall \bigcirc \text{blue}\).
Does there exist ...?

Does there exist a **CTL** formula $\Phi$ s.t. $T_1 \not\models \Phi$ and $T_2 \models \Phi$?

**yes**, as $T_1 \not\models T_2$, e.g., $\Phi = \exists \bigcirc \forall \square \text{blue}$

Does there exist a **LTL** formula $\varphi$ s.t. $T_1 \not\models \varphi$ and $T_2 \models \varphi$?
Does there exist a **CTL** formula $\Phi$ s.t.

$\mathcal{T}_1 \not\models \Phi$ and $\mathcal{T}_2 \models \Phi$ ?

**yes**, as $\mathcal{T}_1 \not\sim \mathcal{T}_2$, e.g., $\Phi = \exists \Diamond \forall \Box blue$

Does there exist a **LTL** formula $\varphi$ s.t.

$\mathcal{T}_1 \not\models \varphi$ and $\mathcal{T}_2 \models \varphi$ ?

**no**, as $\mathcal{T}_1$, $\mathcal{T}_2$ are simulation equivalent
Simulation quotient

Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS.

simulation quotient $\mathcal{T}/\sim$:

transition system that arises from $\mathcal{T}$ by collapsing all simulation equivalent states
Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS. Then:

$$\mathcal{T} / \sim \overset{\text{def}}{=} (S / \sim, Act', \rightarrow_{\sim}, S'_0, AP', L')$$
Let $T = (S, \text{Act}, \to, S_0, \text{AP}, L)$ be a TS. Then:

$$T/\sim \overset{\text{def}}{=} (S/\sim, \text{Act}', \to\sim, S'_0, \text{AP}', L')$$

- state space $S/\sim$ ← set of all simulation equivalence classes
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS. Then:

$$\mathcal{T}/\sim \overset{\text{def}}{=} (S/\sim, \text{Act}', \rightarrow_{\sim}, S_0', AP', L')$$

- state space $S/\sim \leftarrow$ set of all simulation equivalence classes
- initial states: $S_0' = \{[s] : s \in S_0\}$
- labeling: $AP' = AP$ and $L'([s]) = L(s)$

$$[s] = \{s' \in S : s \sim_T s'\}$$
Simulation quotient

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS. Then:

$\mathcal{T}/\sim \overset{\text{def}}{=} (S/\sim, Act', \rightarrow_{\sim}, S'_0, AP', L')$

- state space $S/\sim$ \hspace{2cm} set of all simulation equivalence classes
- initial states: $S'_0 = \{[s] : s \in S_0\}$
- labeling: $AP' = AP$ and $L'([s]) = L(s)$
- transition relation: $\begin{align*}
  s &\rightarrow s' \\
  [s] &\rightarrow_{\sim} [s']
\end{align*}$

action labels: irrelevant
Similarity of $\mathcal{T}$ and $\mathcal{T}/\sim$
Similarity of $\mathcal{T}$ and $\mathcal{T}/\sim$

Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS. Then:

$\mathcal{T}/\sim = (S/\sim, \text{Act}', \rightarrow_{\sim}, S_0', AP, L')$

where the transitions are given by

\[ \frac{s \rightarrow s'}{[s] \rightarrow_{\sim} [s']} \]

$\mathcal{T}$ and $\mathcal{T}/\sim$ are simulation equivalent, i.e.,

$\mathcal{T} \preceq \mathcal{T}/\sim$ and $\mathcal{T}/\sim \preceq \mathcal{T}$

Proof. provide simulations for $(\mathcal{T}, \mathcal{T}/\sim)$ and $(\mathcal{T}/\sim, \mathcal{T})$

simulation for $(\mathcal{T}, \mathcal{T}/\sim)$: $\{(s, [s]) : s \in S\}$

simulation for $(\mathcal{T}/\sim, \mathcal{T})$: ?
Example: simulation quotient

\[ T \]

- \( u_1 \) and \( u_2 \) are simulation equivalent
- \( t_1, t_2, t_3 \) are simulation equivalent
- \( v_1, v_2 \) are simulation equivalent
Example: simulation quotient

\[ T \]

\[ t_1, t_2, t_3 \text{ are simulation equivalent} \]
\[ v_1, v_2 \text{ are simulation equivalent} \]
\[ u_1 \simeq u_2, \quad w \preceq u_1, u_2, \quad \text{but } w \not\simeq u_1, u_2 \]
Example: simulation quotient

$t_1, t_2, t_3$ are simulation equivalent
$v_1, v_2$ are simulation equivalent

$u_1 \sim u_2, \quad w \preceq u_1, u_2, \quad$ but $w \not\succ u_1, u_2$

$s_1 \sim s_2$
Example: simulation quotient

\[ T \]

\[ \mathcal{T} / \sim \]

\[ \{ u_1, u_2 \} \]

\[ \{ v_1, v_2 \} \]

\[ \{ t_1, t_2, t_3 \} \]

\[ \{ s_1, s_2 \} \]

\[ t_1, t_2, t_3 \text{ are simulation equivalent} \]

\[ v_1, v_2 \text{ are simulation equivalent} \]

\[ u_1 \preceq u_2, \quad w \preceq u_1, u_2, \quad \text{but } w \not\sim u_1, u_2 \]

\[ s_1 \preceq s_2 \]
Example: simulation quotient

\[ \mathcal{T} \]

- \( s_1 \) with \( u_1 \) and \( w \) with \( u_2 \)

\[ \mathcal{T}/\sim \]

- \( \{ s_1, s_2 \} \) with \( \{ u_1, u_2 \} \) and \( \{ w \} \)

Simulation for \((\mathcal{T}, \mathcal{T}/\sim)\):

\[ \{ (s, [s]) : s \text{ is a state in } \mathcal{T} \} \]
Example: simulation quotient

\[ \mathcal{T} \]

\[ \mathcal{T}/\simeq \]

Simulation for \((\mathcal{T}, \mathcal{T}/\simeq)\):
\[ \{ (s, [s]) : s \text{ is a state in } \mathcal{T} \} \]

But
\[ \{ ([s], s) : s \text{ is a state in } \mathcal{T} \} \]

Is not a simulation for \((\mathcal{T}/\simeq, \mathcal{T})\)
Example: simulation quotient

\[ T \]

\[ \{ s_1, s_2 \} \]

\[ \{ u_1, u_2 \} \]

\[ \{ w \} \]

show that \( R = \{ ([s], s) : s \text{ is a state in } T \} \) is not a simulation for \((T/\sim, T)\)
Example: simulation quotient

\[ \mathcal{T} \]

\[ \frac{\mathcal{T}}{\sim} \]

show that \( \mathcal{R} = \{ ([s], s) : s \text{ is a state in } \mathcal{T} \} \)

is not a simulation for \((\mathcal{T}/\sim, \mathcal{T})\)

regard \((\{s_1, s_2\}, s_2) \in \mathcal{R}\) and \(\{s_1, s_2\} \rightarrow \sim \{w\}\)
Example: simulation quotient

\[ \mathcal{T} \]

\[ \mathcal{T} / \sim \]

show that \( \mathcal{R} = \{([s], s) : s \text{ is a state in } \mathcal{T} \} \)

is not a simulation for \((\mathcal{T} / \sim, \mathcal{T})\)

regard \(\{s_1, s_2\}\) \(\in \mathcal{R}\) and \(\{s_1, s_2\} \rightarrow \sim \{w\}\)

there is no transition \(s_2 \rightarrow w'\) in \(\mathcal{T}\) s.t. \((\{w\}, w') \in \mathcal{R}\)
Similarity of $\mathcal{T}$ and $\mathcal{T}/\simeq$

Let $\mathcal{T} = (S, \text{Act}, \to, S_0, AP, L)$ be a TS. Then:

$$\mathcal{T}/\simeq = (S/\simeq, \text{Act}', \to\simeq, S'_0, AP, L')$$

where the transitions are given by

\[
\begin{align*}
s & \mapsto s' \\
[s] & \mapsto \simeq [s']
\end{align*}
\]

$\mathcal{T}$ and $\mathcal{T}/\simeq$ are simulation equivalent, i.e.,

$$\mathcal{T} \preceq \mathcal{T}/\simeq \text{ and } \mathcal{T}/\simeq \preceq \mathcal{T}$$

Proof. Provide simulations for $(\mathcal{T}, \mathcal{T}/\simeq)$ and $(\mathcal{T}/\simeq, \mathcal{T})$

Simulation for $(\mathcal{T}, \mathcal{T}/\simeq)$:

$$\{(s, [s]) : s \in S\}$$

Simulation for $(\mathcal{T}/\simeq, \mathcal{T})$: ?
Similarity of \( \mathcal{T} \) and \( \mathcal{T}/\sim \)

Let \( \mathcal{T} = (S, \text{Act}, \rightarrow, S_0, \text{AP}, L) \) be a TS. Then:

\[
\mathcal{T}/\sim = (S/\sim, \text{Act}', \rightarrow_{\sim}, S'_0, \text{AP}, L')
\]

where the transitions are given by \( \frac{s}{[s]} \rightarrow_{\sim} \frac{s'}{[s']} \)

\( \mathcal{T} \) and \( \mathcal{T}/\sim \) are simulation equivalent, i.e.,

\( \mathcal{T} \preceq \mathcal{T}/\sim \) and \( \mathcal{T}/\sim \preceq \mathcal{T} \)

**Proof.** provide simulations for \((\mathcal{T}, \mathcal{T}/\sim)\) and \((\mathcal{T}/\sim, \mathcal{T})\)

simulation for \((\mathcal{T}, \mathcal{T}/\sim)\): \( \{(s, [s]) : s \in S\} \)

simulation for \((\mathcal{T}/\sim, \mathcal{T})\): \( \{([s], t) : s \preceq_{\mathcal{T}} t\} \)